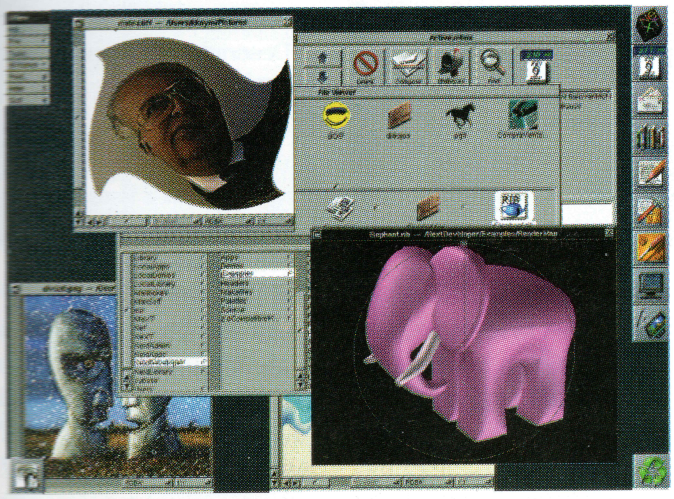
Evaluación del Sistema Operativo NeXTSTEP

en el Grupo Financiero InverMexico



Enrique Ibarra y Gabriel Vergara

En 1993 el Grupo Financiero InverMexico (GFI) tomó la decisión de adoptar la tecnología orientada a objetos para el desarrollo de software. Durante 1994, GFI decidió evaluar las facilidades orientadas a objetos ofrecidas por el sistema operativo NeXTSTEP. Este artículo presenta algunas opiniones y experiencias sobre NeXTSTEP obtenidas por un grupo de trabajo en GFI.

1. Introducción

NeXTSTEP, desarrollado y comercializado por NeXT Computer, Inc., es un sistema operativo basado en UNIX con una interfaz gráfica de usuario (GUI) muy elaborada. NeXTSTEP está construido sobre Mach, un sistema operativo compatible con la version 4.3 del UNIX de Berkeley que fue desarrollado a mediados de los ochentas en la universidad Carnegie Mellon. Cualquier aplicación que corra bajo UNIX 4.3 BSD debe correr sin cambios bajo Mach. Mach aporta extensiones que mejoran los procesos UNIX tradicionales, el manejo de memoria virtual y los mecanismos de comunicación entre procesos. Mach es también un sistema operativo *multithreaded*, es decir, permite la ejecución independiente de varios *procesos ligeros*[1], o *threads*, dentro del mismo proceso UNIX.

Mach aportó una serie de características nuevas que no estaban disponibles en los sistemas UNIX tradicionales. Mach se desarrolló con la intencion de proporcionar una mejor plataforma para equipos con múltiples procesadores y una base sólida para el cómputo distribuido. Mach maneja cuatro abstracciones principales:

- *task* (tarea). Es un ambiente de ejecución que incluye un espacio virtual y paginado de direcciones y acceso protegido a los recursos del sistema. Para que una tarea sea útil, debe tener por lo menos un proceso ligero (*thread*) ejecutándose dentro de ella. En Mach cuando se habla de la comunicación con una tarea, se habla de los medios para comunicarse con un proceso ligero que se esté ejecutando dentro de dicha tarea. Una tarea con un solo proceso ligero ejecutándose dentro de ella es el equivalente en Mach a un proceso UNIX tradicional. Las tareas en Mach son manipuladas esencialmente de la misma forma en que los procesos son manejados por UNIX.
- *thread* (proceso ligero). Es la unidad básica de ejecución. Contiene una pila (*stack*) de ejecución y una cantidad limitada de almacenamiento estático. Comparte el resto de la memoria y recursos con los demás procesos ligeros ejecutándose en la misma tarea. Un proceso ligero sólo se puede ejecutar dentro de una única tarea.
- *port* (puerto). Es un canal de comunicación; una cola de mensajes protegidos por el *kernel*. Sólo una tarea

Enrique Ibarra Anaya, es Doctor en Computación Aplicada a la Ingeniería de la Universidad Carnegie Mellon, de los E.U.A. Actualmente es Subdirector de Desarrollo de Sistemas de Grupo Financiero InverMexico. Sus áreas de interés son: CASE, Metodología Orientada a Objetos y Desarrollo de Aplicaciones Distribuidas. *Tel: 570-7000.* ***Email**: eibarra@gfi.invermexico.com.mx*

Gabriel Vergara, es Ingeniero en Computación del ITAM. Es Analista de Sistemas en Banco Mexicano. Su área de interés es la Programación Orientada a Objetos.

puede recibir mensajes de un puerto, pero todas las tareas que tengan acceso a dicho puerto pueden enviar mensajes.

- *message* (mensaje). Es una colección tipada de datos utilizada en la comunicación entre procesos ligeros.

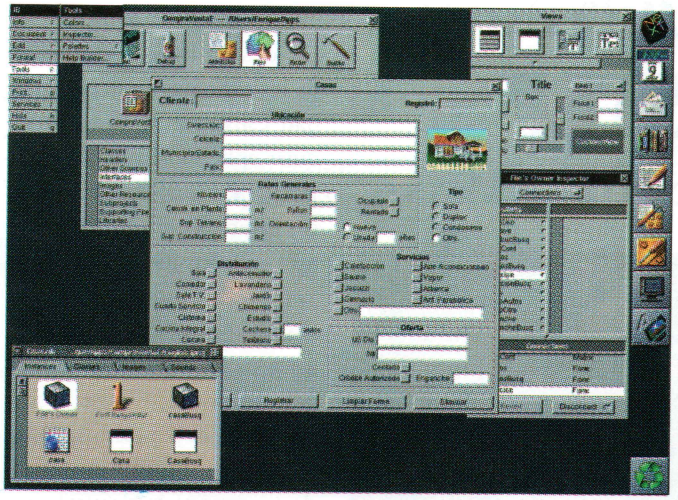
La comunicación entre procesos en UNIX se realiza a través de una variedad de mecanismos como *pipes*, pseudoterminales, señales y *sockets*. Los *sockets* son el mecanismo principal para la comunicación entre procesos UNIX dentro de redes y tienen la desventaja de utilizar los nombres específicos y globales de las máquinas en la red, lo cual no proporciona independencia de localización; tampoco proporcionan mecanismos de seguridad, ya que los datos son transmitidos como un flujo de bytes sin interpretarse por el *kernel*. A través de los puertos y los mensajes, Mach permite la independencia de localización y provee con mecanismos de seguridad y tipado de los datos transmitidos en la comunicación entre procesos.

Las mejoras que Mach aporta a los sistemas UNIX tradicionales están disponibles en NeXTSTEP, haciéndolo una plataforma atractiva para el desarrollo de aplicaciones distribuidas y/o *multithreaded*.

1 **NE**: También hemos usado el término como "hilo de ejecución", multihilos, etc.

Existen muchas publicaciones sobre Mach y sus herramientas para desarrollo de sistemas; algunas de éstas son [1], [2] y [3].

Otra característica especial de NeXTSTEP es que utiliza el mismo mecanismo para producir imágenes en pantalla y en la impresora. Dicho mecanismo, *Display PostScript*, está basado en un superconjunto del lenguaje PostScript estándar. Esto permite obtener impresiones fieles de los gráficos que se ven en pantalla.



En 1989, NeXT lanzó al mercado la primera versión de su sistema operativo que corría en una plataforma de hardware propietaria basada en procesadores Motorola de la serie 68000. El producto originalmente lanzado por NeXT era muy atractivo y vanguardista, sin embargo, para adquirirlo había que comprar de NeXT tanto el hardware como el software. Este hecho desalentó a muchos posibles clientes de NeXT y a desarrolladores de software comercial.

En 1993 Steve Jobs, el fundador de NeXT, decidió abandonar la fabricación de hardware para concentrarse en software orientado a objetos. Esta decisión iba dirigida a poder ofrecer su sistema operativo, NeXTSTEP, en multiples plataformas de hardware. En la actualidad, NeXT ofrece dos productos:

- *NeXTSTEP*. Sistema operativo que de momento está disponible para computadoras basadas en el procesador Intel i486 o Pentium, y para computadoras HP basadas en el procesador PA-RISC.

- *OpenStep*. Conjunto de APIs que estará disponible en plataformas Sun y Digital.

NeXT argumenta que las aplicaciones NeXTSTEP y OpenStep podrán ser migradas a cualquier plataforma que soporte uno de estos dos productos con compatiblidad total de código fuente. NeXT espera aumentar su presencia en el mercado apoyándose en la anterior compatibilidad y en alianzas de comercialización hechas con sus anteriores rivales: Sun, HP y DEC. Esta estrategia parece estar dándole frutos a NeXT ya que, según sus datos, en 1994 alcanzaron más de 100,000 usuarios de NeXTSTEP, más del doble de los que tenían a finales de 1993.

2. Descripción del Ambiente NeXTSTEP

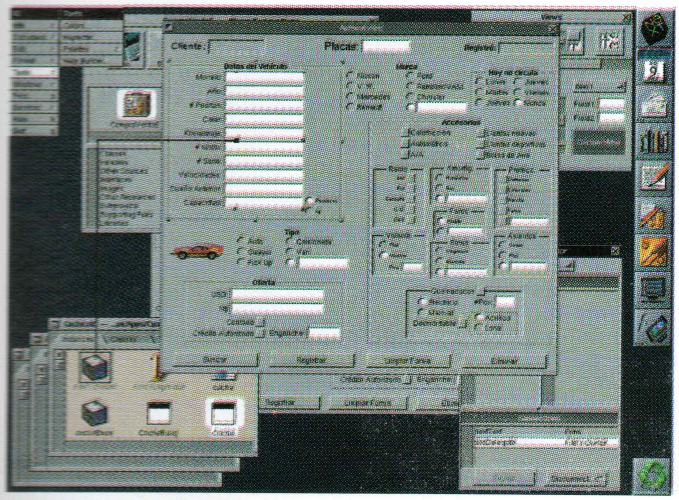
NeXTSTEP es una plataforma dirigida al desarrollo rápido de sistemas a la medida en un ámbito corporativo global y con filosofía cliente-servidor. NeXTSTEP es una plataforma diseñada para el desarrollo de aplicaciones que hagan el papel de cliente. Por su excelente ambiente gráfico, NeXTSTEP es una plataforma que inherentemente se presta para el desarrollo de aplicaciones que directamente interactúen con personas.

Las principales razones por las cuales decidimos evaluar NeXTSTEP en GFI son:

- las posibilidades de prototipado rápido;
- el uso de vanguardia de la tecnología orientada a objetos; y
- su ambiente altamente gráfico.

MeXT, los distribuidores de NeXT en México, nos facilitaron una copia de evaluación de NeXTSTEP.

Instalamos una copia de NeXTSTEP versión 3.2 para desarrollador en una computadora personal HP modelo 486/XM con 40 Mb de RAM, 450 Mb de disco duro, 2 Mb de memoria de video y monitor a color de diecisiete pulgadas. Existen dos versiones de NeXTSTEP: una para desarrolladores de sistemas, y otra para usuarios finales. La versión para desarrolladores de sistemas incluye las herramientas necesarias para la construcción de aplicaciones NeXTSTEP.



Hay que destacar que NeXTSTEP no se puede instalar en cualquier PC; es necesario tener una computadora con amplias capacidades de memoria y de video para que sea cómodo de usar. Para equipos basados en el procesador i486 recomendamos tener cuando menos 32 Mb de RAM y 2 Mb de memoria de video; la cantidad de disco depende de la versión de NeXTSTEP que se vaya a manejar. También cabe mencionar que NeXTSTEP es un sistema operativo caro si se compara con sus principales competidores: Windows NT, OS/2 y las diversas versiones de UNIX disponibles en el mercado (Solaris, Santa Cruz Operation, UnixWare, etc.).

Lo primero que llama la atención al utilizar NeXTSTEP es lo bien logrado de su ambiente gráfico. Es una plataforma que al usuario le resulta muy amigable y agradable. El mecanismo de interacción con menús es diferente al de cualquier otro GUI, sin embargo, es fácil acostumbrarse a su uso.

Una vez instalado el sistema operativo, la máquina con NeXTSTEP se integró en nuestra red local como una computadora UNIX más, capaz de interactuar con el resto de nuestros equipos UNIX (HP, IBM y Sun). Pudimos sin ningún problema establecer sesiones *telnet* y *ftp* hacia y desde la máquina con NeXTSTEP. Pudimos también instalarle a la máquina NeXTSTEP, sin problemas, las herramientas GNU que utilizamos en el resto de la máquinas UNIX. Después de extensas pruebas comparativas nos percatamos que el rendimiento de la máquina con NeXTSTEP (con la configuración descrita anteriormente) era equivalente (muy ligeramente menor) al de una *workstation* HP 715/33 con 32 Mb de RAM. Lo que no resultó fácil fue la instalación de una impresora en la red, ya que los programas visuales de administración de NeXTSTEP suponen que la impresora se va a conectar directamente a un puerto local de la computadora. Este es un detalle que se podría mejorar en futuras versiones.

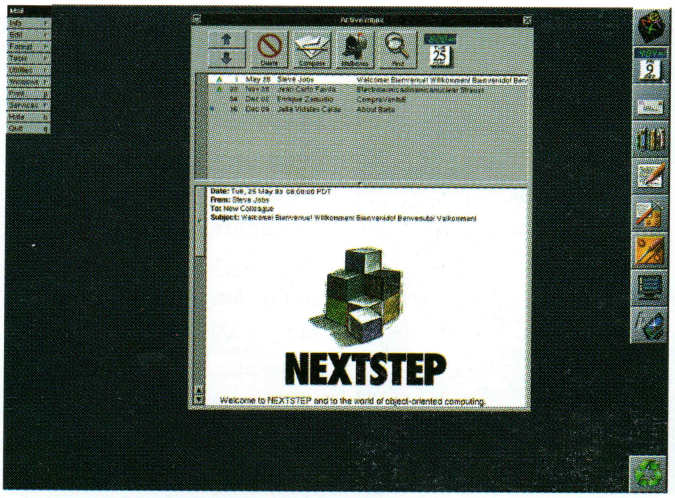
El *Workspace Manager* de NeXTSTEP es un programa que le permite al usuario manejar y personalizar el ambiente de trabajo y de la pantalla; es la interfaz gráfica hacia el sistema UNIX. El *Workspace Manager* es similar al *Macintosh Finder* y al *File Manager* y *Program Manager* de Microsoft Windows en el sentido de que permite iniciar programas y manejar el sistema de archivos. A diferencia de los programas anteriores, el *Workspace Manager* permite

Lo que primero llama la atención al utilizar NeXTSTEP es lo bien logrado de su ambiente gráfico. Es una plataforma que al usuario le resulta muy amigable y agradable.

explotar las posibilidades de una maquinaria subyacente mucho más poderosa siendo al mismo tiempo muy fácil de utilizar. El administrador de archivos de NeXTSTEP nos pareció la mejor herramienta que actualmente existe para manipular archivos en un sistema jerárquico de archivos.

La versión de desarrollador de NeXTSTEP ofrece una serie de herramientas y bibliotecas para el rápido desarrollo y prueba de aplicaciones. La principal herramienta que se utiliza en el desarrollo de una aplicación es *Project Builder*, una herramienta que administra todos los componentes de una aplicación y da acceso al resto de las herramientas de desarrollo que se utilizan para crear, modificar y/o probar dichos componentes. *Project Builder* se utiliza a lo largo de todo el proceso de desarrollo, desde la creación de los bloques básicos para una nueva aplicación, hasta la instalación de la aplicación una vez que ha sido terminada y probada.

Otra herramienta fundamental que se utliza en la creación de una aplicación NeXTSTEP es el *Interface Builder*, que permite construir gráficamente la interfaz de una aplicación, es decir, las ventanas que utilizará la aplicación y los controles gráficos contenidos en cada ventana (botones, deslizadores, menús, etc.). Con *Interface Builder*, se crean y editan interactivamente los objetos gráficos de una interfaz de usuario. Estos objetos se toman directamente de paletas que contienen objetos NeXTSTEP que representan controles gráficos y se arrastran directamente a la interfaz que estamos construyendo. Una vez que un objeto se ha incorporado en



nuestra aplicación se pueden alterar sus características específicas, por ejemplo, se puede editar el título de un botón o ajustar los límites superior e inferior de un deslizador, entre otras cosas.

Una vez que los objetos gráficos de una interfaz han sido reunidos y editados, *Interface Builder* permite establecer gráficamente la interacción entre dichos objetos conectándolos entre sí y especificando la interacción entre ellos. Por ejemplo, un botón se puede conectar a una ventana para que cuando el botón sea accionado, la ventana desaparezca de la pantalla. Aún antes de escribir una sola línea de código, se puede probar el funcionamiento de la interfaz gráfica. La interfaz diseñada con *Interface Builder* se almacena en un archivo denominado *nib* que es utilizado por la aplicación en tiempo de ejecución.

Los objetos gráficos manejados por *Interface Builder* son controles de uso general que se utilizan en las interfaces gráficas. El código con la lógica propietaria de la aplicación que está siendo desarrollada se encapsula dentro de objetos que pertenecen a clases programadas en Objective-C y que interactúan con algunos de los objetos gráficos de la interfaz.

Otras de las herramientas de desarrollo que ofrece NeXTSTEP son:

- *Compilador GNU CC*. El compilador de NeXTSTEP está basado en el compilador GNU CC, producido por la FSF (Free Software Foundation). El compilador ha sido modificado por NeXT para trabajar con el lenguaje Objective-C, además de poder compilar ANSI C y C++.
- *gdb*. Es el depurador (*debugger*) de GNU, producido por la FSF. *gdb* es un depurador muy poderoso; NeXT modificó *gdb* para manejar código Objective-C y le agregó una interfaz gráfica que puede ser invocada desde *Project Builder*. *gdb* también puede ser invocado desde una línea de comando en un *shell*, desde *Edit* o desde *emacs*.
- *Malloc Debugger*. Herramienta que mide el uso que un programa hace de la memoria dinámica. Se puede utilizar para medir toda la memoria reservada por una aplicación o para medir la memoria reservada desde un cierto instante. También provee un detector para localizar las fugas de memoria.
- *Yap*. Visualizador interactivo de PostScript para desarrolladores que deseen escribir y probar código PostScript.
- *Edit*. Editor de texto que maneja archivos ASCII y RTF (Rich Text Format).
- *Terminal*. Emulador de terminales VT100 para ejecutar comandos y programas UNIX tradicionales desde un shell.

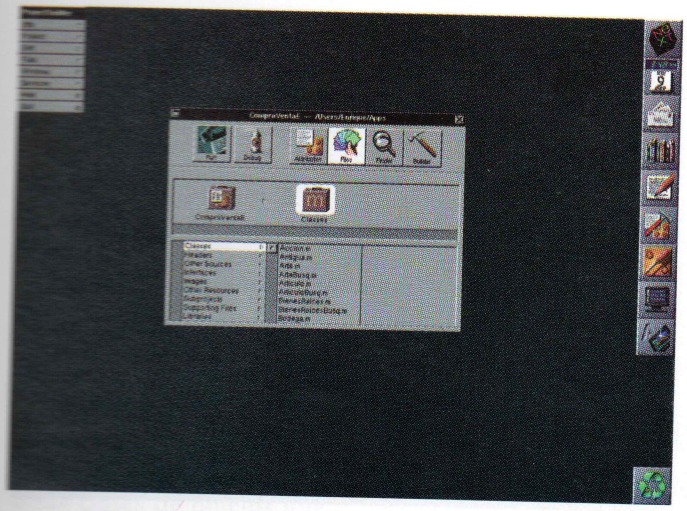
Además de probar las herramientas de desarrollo que son

parte de NeXTSTEP tuvimos la oportunidad de probar *Co-Xist*, un programa que se vende por separado y que permite ejecutar aplicaciones X11R5 en NeXTSTEP. *Co-Xist* es un producto que le añade a NeXTSTEP una versatilidad enorme, ya que permite no sólo ejecutar aplicaciones X11 en la máquina NeXTSTEP local sino que permite visualizar en NeXTSTEP aplicaciones X11 que se estén ejecutando remotamente en otras máquinas UNIX.

Una interesante aplicación NeXTSTEP que no pudimos probar es *Enterprise Objects Framework* (EOF). EOF es una herramienta dirigida a permitir el rápido desarrollo de aplicaciones que interactúen con bases de datos relacionales. EOF le permite a los desarrolladores construir clases de objetos que combinen la lógica de la aplicación y el almacenamiento permanente de objetos en los manejadores relacionales de bases de datos Oracle o Sybase. La falta de un adaptador de EOF para Ingres nos impidió evaluar esta herramienta.

3. Desarrollo de una Aplicación en NeXTSTEP

Lo que más nos interesaba evaluar de NeXTSTEP en GFI era el proceso de desarrollo de una nueva aplicación y el esfuerzo involucrado en dicho desarrollo. Escogimos en GFI una aplicación que ya estaba analizada y las clases derivadas de dicho análisis ya habían sido implementadas en C++ y probadas extensamente; de hecho, estas clases son parte de una aplicación que ya estaba en uso y que consiste en un proceso servidor. Lo que nos faltaba era desarrollar una buena interfaz gráfica para construir, borrar y modificar gráficamente objetos de las clases ya implementadas, ya que hasta entonces se creaban y editaban a través de programas C++. Los objetos de dichas clases representan conceptos complejos y era necesario construir una interfaz gráfica que permitiera la fácil creación, edición y manipulación de dichos conceptos.



Para contrastar el esfuerzo de desarrollo en NeXTSTEP decidimos desarrollar en paralelo la misma interfaz gráfica en una máquina UNIX utilizando C++ y la biblioteca de Motif versión 1.2. El desarrollo en Motif se hizo sin la ayuda de una herramienta gráfica, sólo se contaba con el compilador C++ y la biblioteca de Motif. En este ejercicio participaron dos analistas con experiencias equivalentes en programación orientada a objetos y en el uso del lenguaje C++. Uno de estos analistas iba a ser responsable de desarrollar la interfaz en NeXTSTEP; el otro sería responsable de desarrollar la misma interfaz utilizando Motif y C++. El diseño de la interfaz fue un trabajo conjunto de estos dos analistas.

El analista responsable del desarrollo en C++ y Motif tenía un ligero conocimiento de programación con X11 y Motif y tuvo que profundizarlo a través del estudio de varios libros ([4], [5]). El analista responsable del desarrollo en NeXTSTEP invirtió una buena cantidad de tiempo familiarizándose con el sistema operativo y las herramientas de desarrollo. Una de las diferencias principales entre am-

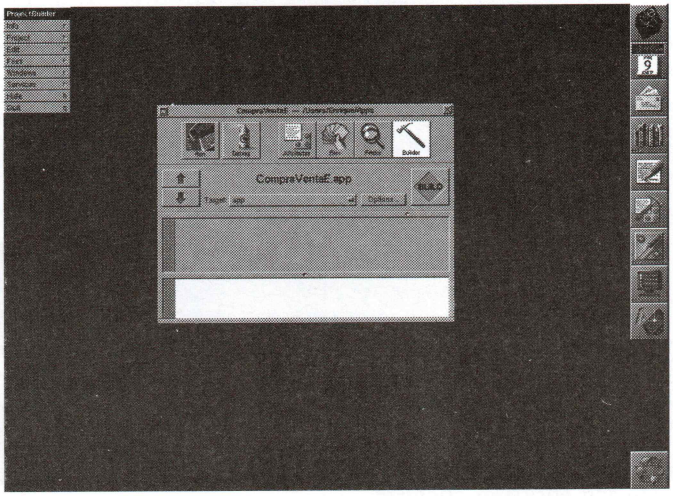
La versión de desarrollador de NeXTSTEP ofrece una serie de herramientas y bibliotecas para el rápido desarrollo y prueba de aplicaciones en ambiente cliente-servidor.

bos desarrollos es que la interfaz en NeXTSTEP tenía que ser programada en Objective-C, lenguaje orientado a objetos que es un superconjunto de ANSI-C. Las clases de objetos que representan controles para interfaces gráficas en NeXTSTEP están programadas en Objective-C. NeXT argumenta que por sus características y facilidades de ligado dinámico, Objective-C es mejor opción que C++ y por eso lo escogieron como su lenguaje de desarrollo. Desafortunadamente para NeXT, Objective-C es un lenguaje que no alcanzó ni siquiera una fracción del éxito que ha tenido C++; este último lenguaje se ha convertido en un estándar *de facto* en la programación orientada a objetos. Debido a su anterior experiencia en análisis orientado a objetos y en programación C++, el aprendizaje de Objective-C no fue mayor problema para el analista responsable del desarrollo en NeXTSTEP.

El esfuerzo para desarrollar interfaces gráficas en NeXTSTEP es mínimo, sobre todo si se compara con el esfuerzo equivalente utilizando C++ (o ANSI-C) y la biblioteca de Motif directamente.

Como se mencionó anteriormente, el diseño de la interfaz fue un trabajo conjunto de los dos analistas. Para realizar este diseño utilizaron el *Interface Builder* de NeXTSTEP. *Interface Builder* es una excelente herramienta para diseñar rápidamente prototipos de interfaces de usuario. Hay que destacar que Interface Builder no es una simple herramienta de prototipado rápido; se utiliza en el diseño y construcción de la interfaz definitiva de las aplicaciones NeXTSTEP, sin embargo, se puede utilizar perfectamente para el diseño y prueba de prototipos.

Gracias a las facilidades que ofrece *Display PostScript* en NeXTSTEP, cada una de las ventanas de la interfaz de la aplicación se imprimió en papel y al final se formó un documento con la descripción gráfica de los componentes individuales de la interfaz a desarrollarse. Este documento le sirvió al desarrollador en Motif como base para reproducir el diseño de la interfaz utilizando los componentes gráficos de Motif.



El esfuerzo realizado en la implementación del cascarón de la interfaz gráfica fue radicalmente menor en NeXTSTEP que en C++ y Motif. Por cascarón nos referimos a la implementación de la interfaz gráfica sin interactuar aún con los objetos del dominio de la aplicación. Pudimos constatar que el esfuerzo para desarrollar interfaces gráficas en NeXTSTEP es mínimo, sobre todo si lo comparamos con el esfuerzo equivalente utilizando C++ (o ANSI-C) y la biblioteca de Motif directamente.

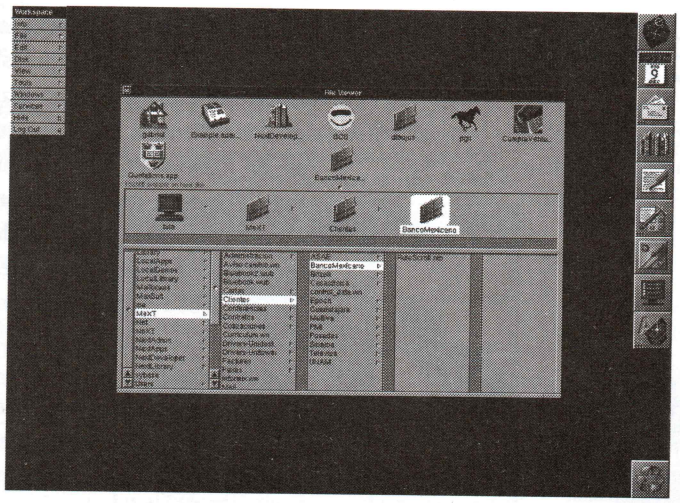
Hasta este punto, todo indicaba que NeXTSTEP era la plataforma ideal para realizar nuestras aplicaciones gráficas en ambiente UNIX, sin embargo, surgieron problemas: no pudimos agregarle a la aplicación NeXTSTEP que estabamos desarrollando el código fuente de nuestras clases escritas en C++. La implementación de algunas de estas clases está basada en el uso de *templates* (clases paramétricas) y el compilador de NeXTSTEP versión 3.2 no reconoce los *templates* de C++. Esto nos resultó muy extraño ya que la versión pública de gcc, el compilador ANSI-C y C++ de GNU, soporta desde hace tiempo el uso de *templates* C++. Una aplicación NeXT-

STEP puede estar implementada, según la documentación de NeXTSTEP, en Objective-C junto con ANSI-C y/o C++, el problema es que no está soportado aún el uso de *templates* de C++. A nuestro juicio, la anterior es una limitante importante.

Una solución al problema con el que nos topamos era reimplementar las clases C++ que hacían uso de *templates* para volverlas a construir sin el uso de *templates*. Debido a la extensa reutilización de dichas clases, el esfuerzo de reimplementación no se justificaba. La reutilización es parte de nuestra filosofía de desarrollo en GFI. La implementación de nuevas clases de objetos se apoya por lo general en la funcionalidad ofrecida por otras clases de objetos que han sido previamente implementadas y probadas.

Debido al problema anteriormente descrito, decidimos abandonar el desarrollo en NeXTSTEP. El desarrollo en C++ y Motif continuó y nos llevo cuatro meses completar la aplicación.

Otro de los inconvenientes de NeXTSTEP es la poca disponibilidad de software comercial. Por ejemplo, en GFI utilizamos Tuxedo como monitor de transacciones UNIX. La gran mayoría de las aplicaciones cliente que desarrollamos en GFI hacen uso de transacciones ofrecidas por servidores Tuxedo; no existe de momento una versión cliente de Tuxedo para NeXTSTEP. En GFI tenemos una infraestructura importante de servidores Tuxedo los cuales realizan las transacciones que soportan la operación de nuestra institución; de momento estas transacciones no pueden ser accedidas por aplicaciones NeXTSTEP, lo cual limita de forma importante el espectro de aplicaciones de NeXTSTEP en GFI.



4. Conclusiones

NeXTSTEP es un sistema operativo que ofrece una base sólida para la rápida construcción de aplicaciones personalizadas utilizando técnicas de orientación a objetos. Es de aplaudirse lo bien logrado de su ambiente gráfico y su facilidad de uso. NeXTSTEP ofrece adicionalmente todas las ventajas de Mach: posibilidad de construir aplicaciones *multithreaded*, facilidades para el cómputo distribuido, etc. Desde el punto de vista de desarrollo de nuevas aplicaciones, las herramientas que ofrece NeXTSTEP cumplen su cometido: permitir el rápido desarrollo de aplicaciones con interfaces gráficas.

Sin embargo, NeXTSTEP tiene aún poca participación en el mercado de los sistemas operativos y es difícil encontrar versiones del software comercial más popular para NeXTSTEP. En el caso de GFI, Tuxedo y el manejador de base de datos Ingres son dos de las herramientas que más utilizamos en la construcción de aplicaciones. NeXTSTEP de momento no puede interoperar con ninguna de estas dos herramientas, lo cual descalifica a NeXTSTEP como plataforma de desarrollo útil para GFI.

El fallo que más nos afectó fue la imposibilidad del compilador de NeXTSTEP para poder compilar código C++ que haga uso de *templates*. NeXT debería corregir esta deficiencia cuanto antes ya que la mayoría de las aplicaciones orientadas a objetos están siendo desarrolladas en C++ y el uso de *templates* se está extendiendo entre los programadores. MeXT nos comentó que la próxima versión de NeXTSTEP tendrá resuelto el problema anterior. También argumentan que pronto será posible la interoperabilidad de NeXTSTEP con Tuxedo e Ingres. Si estas barreras son eliminadas, NeXTSTEP se convertiría seguramente en nuestra plataforma de elección para aplicaciones gráficas en UNIX.

Artículo Arbitrado

Referencias

[1] Acetta, M.J.; Baron, R.V.; Bolosky, W.; Golub, D.B.; Rashid, R.F.; Tevanian, A., and Young, M.W., "Mach: A new kernel foundation for Unix development", *Proceedings of Summer Usenix*, July, 1986.

[2] Eppinger, J.L.; Mummert, L.B., and Spector, A.Z., "Camelot and Avalon. A Distributed Transaction Facility", Morgan Kauffman, 1991.

[3] Walmer, L.R., and Thompson, M.R., "A programmer's guide to the Mach system calls", Technical Report, Department of Computer Science, Carnegie Mellon University, December, 1989.

[4] Young, D.A., "Object Oriented Programming with C++ and OSF/Motif", Prentice Hall, 1992.

[5] Young, D.A., "The X Window System Programming and Applications with X11 OSF Motif Edition", Prentice Hall, 2nd. Edition, 1994.